Trabajo Fin de Grado

Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Análisis de la robustez del protocolo SpaceWire frente a SEUs mediante FT-UNSHADES2

Autor: Gonzalo Federico Uncal

Tutores: Hipólito Guzmán Miranda

Miguel Ángel Aguirre Echanove

Dpto. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2018

Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Análisis de la robustez del protocolo SpaceWire frente a SEUs mediante FT-UNSHADES2

Autor:
Gonzalo Federico Uncal

Tutores:
Hipólito Guzmán Miranda
Profesor Contratado Doctor

Miguel Ángel Aguirre Echanove
Catedrático de Universidad

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2018

Trabajo Fin de Grado: Análisis de la robustez del protocolo SpaceWire frente a SEUs mediante FT-UNSHADES2

Autor: Gonzalo Federico Uncal

Tutores: Hipólito Guzmán Miranda
Miguel Ángel Aguirre Echanove

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

A mi familia

A mis amigos

A todos los que han contribuido en mi formación

Agradecimientos

Quiero agradecer a Daniel Acerbi, profesor e ingeniero durante mis estudios universitarios en mi país natal, por introducirme en el fantástico mundo de la electrónica digital, y por marcar una dirección en el camino a tomar en el ámbito profesional.

He de agradecer también a los catedráticos Miguel Ángel Aguirre Echanove y Fernando Muñoz Chavero y el Doctor Hipólito Guzmán Miranda por la amplia y de gran importancia formación que han impartido tanto a mí como a muchos otros alumnos y por definir aún más dicho camino a seguir.

Especialmente quiero agradecer a Miguel Ángel Aguirre Echanove, tutor de este TFG, por su dedicación, buen trato y por confiarme este trabajo, aunque por motivos de salud no haya podido continuar con sus labores. Por ello, agradezco mucho a Hipólito Guzmán Miranda, quien tomó su lugar como tutor, por su apoyo y su ayuda al resolver dudas y problemas que han surgido, respondiéndome a correos cuando le era posible, aún estando desbordado de trabajo. Ambos grandes profesionales y personas.

Obviamente debo agradecer a mi familia y amigos por todo el apoyo brindado durante esta etapa que, a pesar de algún altibajo, está a punto de finalizar.

Gracias a todos.

Gonzalo Federico Uncal

Sevilla, Septiembre de 2018

Resumen

El proyecto realizado consiste en el análisis de un enlace de comunicación compuesto por un emisor y un receptor, utilizando el protocolo SpaceWire para la transmisión de datos, frente a SEUs. El enlace está completamente diseñado en VHDL y posteriormente se implementará en la plataforma de inyecciones de fallos FT-UNSHADES 2.

FT-UNSHADES 2 permite poner a prueba un diseño ante un entorno de radiación ionizante que pueda provocar cambios en los bits de configuración y en las memorias. La plataforma utiliza FPGAs VIRTEX-5 debido a su facilidad de poder realizar cambios en su configuración.

El objetivo de utilizar esta plataforma es obtener resultados de inyecciones de fallos, que emulan la radiación ionizante en el enlace SpaceWire, luego tratar estos resultados en Matlab para clasificar los fallos, es decir, si los datos transferidos son erróneos, si ocurre un error en el enlace que haga que deje de transmitir, etc. y poder llegar a una conclusión sobre el funcionamiento del protocolo en entornos radiantes que pueden ser encontrados durante misiones espaciales.

Abstract

The project consists in the analysis of a communication link composed of an emitter and a receiver, using the SpaceWire protocol for data transmission, against SEUs. The link is completely designed in VHDL and it will be implemented in the FT-UNSHADES2 fault injection platform.

FT-UNSHADES2 allows you to test a design in a ionizing radiation environment wich can cause bit-flips in configuration bits and memories. The platform uses FPGAs Virtex-5 models due to its ease of being able to make changes to its configuration.

The objective of using this platform is to obtain results of fault injections, which emulate ionizing radiation in the SpaceWire link, then analyze these results in Matlab to classify the faults, that is, if the transferred data are erroneous, if an error occurs in the link that causes it to stop transmitting, etc. and be able to come to a conclusion about the operation of the protocol in radiative environments that can be found during space missions.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

SEU	Single Event Upset
GCR	Galactic Cosmic Ray
eV	Electronvoltio
NASA	National Aeronautics and Space Administration
JAXA	Japan Aerospace Exploration Agency
ESA	European Space Agency
TID	Total Ionizing Dose
CMOS	Complementary Metal-Oxide-Semiconductor
SEE	Single Event Events
SEL	Single Event Latchup
SEB	Single Event Burnout
LET	Linear Energy Transfer
LETth	Linear Energy Transfer Threshold
CRC	Cyclic Redundance Check
EOP	End of Packet
EEP	Error End of Packet
MPO	Mercury Planet Orbiter
MMO	Mercury Magnetospheric Orbiter
MTM	Mercury Transfer Module
LDPC	Low Density Parity Check
RAM	Random Access Memory
FPGA	Field-Programmable Gate Array
TCL	Tool Command Language
MSB	Most Significant Bit
VHDL	Very High Speed Integrated Circuit Hardware Description Language
SoC	System-on-Chip
SoCWire	System-on-Chip Wire
NoC	Network-on-Chip
SRAM	Static Random Access Memory

1 INTRODUCCIÓN

A través de la historia, uno de los principales objetivos del ser humano ha sido la búsqueda de nuevos territorios por motivos como la obtención más de poder, el aumento del conocimiento o la supervivencia. Esto lo ha obligado a tener que adaptarse para poder enfrentarse a lo desconocido y así permitir que próximas generaciones puedan continuar con este objetivo.

El humano no tardó mucho en interesarse por lo que hubiera más allá de la superficie terrestre, pero, sin la capacidad para viajar y explorar todo lo que le rodeaba, solo podía estudiar y tratar de comprender ese, a simple vista, inmenso territorio desconocido y los fenómenos que ocurrían en el y como nos afectaba. No fue hasta mediados del siglo XX cuando se comenzaron a realizar exploraciones con naves no tripuladas con el fin de estudiar, entre otros, la radiación cósmica.

Hoy en día la tecnología ha avanzado lo suficiente como para tener en funcionamiento multiples naves tripuladas y no tripuladas con el objetivo de exploración y obtención de datos, comunicación vía satélite, etc. Esto ha sido posible gracias al previo estudio de los fenómenos que afectan a dicha instrumentación y a los humanos que se encuentren operándola, y la mejora y adaptación de la tecnología para poder operar en dichas condiciones.

La radiación cósmica es uno de los fenómenos que, sobre todo en el espacio, es un reto para los investigadores debido a que las partículas que componen esta radiación son ionizantes y afectan a los dispositivos electrónicos utilizados en misiones espaciales. La vulnerabilidad a la radiación ionizante no es la misma para toda la electrónica, la tecnología CMOS por ejemplo, es susceptible a los efectos de esta radiación por lo que se deben buscar maneras de proteger la electrónica para que puedan desarrollar sus funciones en estos entornos.

Las misiones espaciales tienen objetivos cada vez más ambiciosos, los dispositivos de almacenamiento deben prepararse para almacenar cada vez más cantidad de datos, los datos obtenidos cada vez son más pesados y con más resolución y el objetivo es que las velocidades de transmisión de datos sean cada vez más altas. Para poder cumplir estos objetivos hace falta que también se mejore la mitigación de los efectos de la radiación ionizante a nivel de hardware y software.

Las FPGAs (Field-Programable Gate Array) son muy utilizadas en misiones espaciales debido a que pueden ser programadas para desarrollar una función lógica especifica y pueden encargarse de realizar las tareas de procesamiento de datos, entre otras. Muchos son los modelos de FPGAs creados a lo largo del tiempo y no todos son aptos para su uso en el espacio, la primera FPGA adaptada para su uso en misiones espaciales fue la basada en antifusibles. En esta FPGA los datos de configuración son guardados en antifusibles una vez y son permanentes, esta partícularidad la hace inmune a la radiación ionizante, pero pierde capacidad de funciones logicas y velocidad de operación.

Las nuevas FPGAs empleadas en misiones espaciales se basan en la tecnología SRAM. Estas utilizan memorias SRAM para el almacenamiento de datos de configuración y pueden realizar una gran cantidad de

operaciones a una frecuencia muy alta, aunque sus memorias de configuración y almacenamiento son susceptibles a los efectos de la radiación, por lo que para poder utilizarse en misiones espaciales deben contar con métodos de protección, a nivel de hardware y software, para una mejor mitigación de los efectos de la radiación ionizante.

1.1 Alcance

El objetivo del trabajo es estudiar el efecto de la radiación ionizante en un enlace de comunicación que utiliza el protocolo SpaceWire, mediante el uso de la plataforma desarrollada por la Universidad de Sevilla FT-UNSHADES2, que puede simular un entorno de radiación ionizante mediante la inyección automatizada de fallos en puntos sensibles del enlace SpaceWire. Dicha red será un enlace emisor-receptor entre dos nodos SpaceWire que simularán la transmisión de datos que podría darse en una misión espacial.

La red SpaceWire será diseñada completamente en VHDL utilizando el software ISE 14.7 de Xilinx y sintetizada para la FPGA Virtex XC5VFX70T, la cual es la utilizada por la plataforma FTU2. Posteriormente el diseño será implementado en la plataforma, se deberá comprobar el correcto funcionamiento del protocolo antes de proceder a realizar una campaña de inyecciones.

La campaña con la que se obtendrian los mejores resultados sería una campaña exhaustiva, ya que esta inyecta fallos en cada ciclo y bit posible, pero, dado que la cantidad de bits que conforman los registros de usuario es aproximadamente 1000 y que el rango de ciclos (se explicará en el CAPITULO 5) en los que se inyectará es de aproximadamente 900, se realizará una campaña aleatoria con una cantidad de 50000 ejecuciones que devolverán datos suficientes para analizar el comportamiento del protocolo frente a los efectos de la radiación ionizante.

Los resultados obtenidos de la campaña de inyecciones serán tratados mediante el uso de Matlab con el objetivo de crear una clasificación de fallos generados por la simulación de los efectos de la radiación.

Mediante estos resultados se podrá comprobar la gran utilidad del uso de las herramientas que ofrece la plataforma FT-UNSHADES2 y a partir de esta, un estudio sobre la mitigación de los efectos de la radiación ionizante sobre un estándar de comunicación actualmente utilizado por dispositivos electrónicos en misiones espaciales.

2 RADIACIÓN IONIZANTE EN ENTORNOS ESPACIALES

2.1 Radiación cósmica

Esta radiación está conformada por tres tipos de radiación [1]: partículas atrapadas en el campo magnético de la tierra, partículas emitidas al espacio durante erupciones solares y rayos cósmicos galácticos.

2.1.1 Rayos cósmicos galácticos

Los componentes que conforman la radiación cósmica galáctica (GCR-Galactic Cosmic Radiation) son protones energéticos e iones pesados con números atómicos pares, que son más abundantes que aquellos con números atómicos impares. La relativa abundancia de cada tipo de partícula generalmente decrece con el aumento del número atómico. Al colisionar con nucleos de oxígeno y nitrógeno, particulas que conforman la atmósfera de la tierra, estos se fragmentan en partículas más ligeras y simultáneamente se emiten neutrones.

Estos rayos vienen desde fuera del sistema solar, son más escasas que las partículas solares provenientes de eventos solares y mucho más escasas que las partículas atrapadas en órbitas bajas de la tierra, pero alcanzan valores de energía mucho más altos, del orden de TeV [2].

Los rayos cósmicos galácticos son relativamente constantes en términos de tipos de distribución de partículas y energías a travez del tiempo, pero estos decrecen en intensidad con un factor de 10 durante los eventos solares debido a que el incremento de la energía emitida por el sol produce un incremento del campo magnético interplanetario que desvía una gran fracción de los rayos cósmicos galácticos.

2.1.2 Partículas solares

Los eventos solares producen una intensificación sustancial de la mayoría de las partículas energéticas (incluyendo protones con energías máximas del orden de GeV, iones pesados cuya energía máxima es del orden de GeV [2] y electrones) que emanan del sol. De estas partículas emitidas por el sol, las más peligrosas para una misión espacial son mayormente los protones y, en menor medida, iones pesados, que son relativamente menos abundantes que en el caso de los rayos cósmicos.

La intensidad de estas partículas puede variar en varios órdenes de magnitud, esta variación en la intensidad se traduce en una significante incertidumbre en el riesgo que esto supone para la misión, aunque ese riesgo disminuye cuando la actividad solar se va acercando al mínimo del ciclo solar.

En la Figura 2.1 puede observarse un registro de las emisiones solares a travez de varios ciclos con el paso del tiempo.

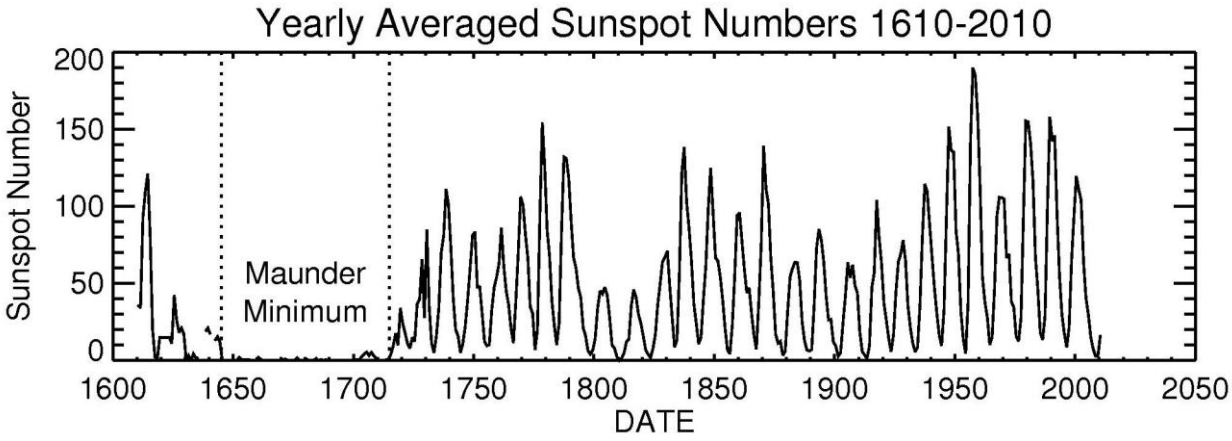

Figura 2.1 – Emisiones solares registradas a travez del paso del tiempo [3].

2.1.3 Cinturones de Van Allen

En 1958 el Doctor James Van Allen descubrió dos "cinturones" de radiación que cubren la tierra, el cinturón interior y el exterior. El cinturón interior está compuesto por protones y electrones, mientras que el exterior principalmente se compone de electrones. Estas partículas están atrapadas en estos cinturones debido a la fuerza magnetosférica, pero estas están en constante movimiento, el cual es muy complejo. La energía máxima de las mismas puede alcanzar ordenes de, decenas de MeV en los electrones y centenas de Mev en el caso de protones e iones pesados [2].

El contenido de partículas de los cinturones cambia lentamente en el tiempo cuando más partículas son inyectadas en los cinturones o cuando las partículas consiguen escapar de la fuerza magnética de estos. Estos cambios son, principalmente, producto de los cambios en la fuerza magnetosférica producidos por la actividad solar. Las actividades humanas, como las pruebas nucleares, inyectan una gran cantidad de partículas en la atmósfera e impactan en la composición de los cinturones.

En el 2012 la NASA, mediante el uso de detectores de partículas durante la misión Radiation Belt Storm Probes (RBSP), descubrió que hay un tercer cinturón entre el interior y el exterior, pero tuvo una duración temporal de un mes aproximadamente, pero apareció nuevamente cuando la actividad solar aumentó.

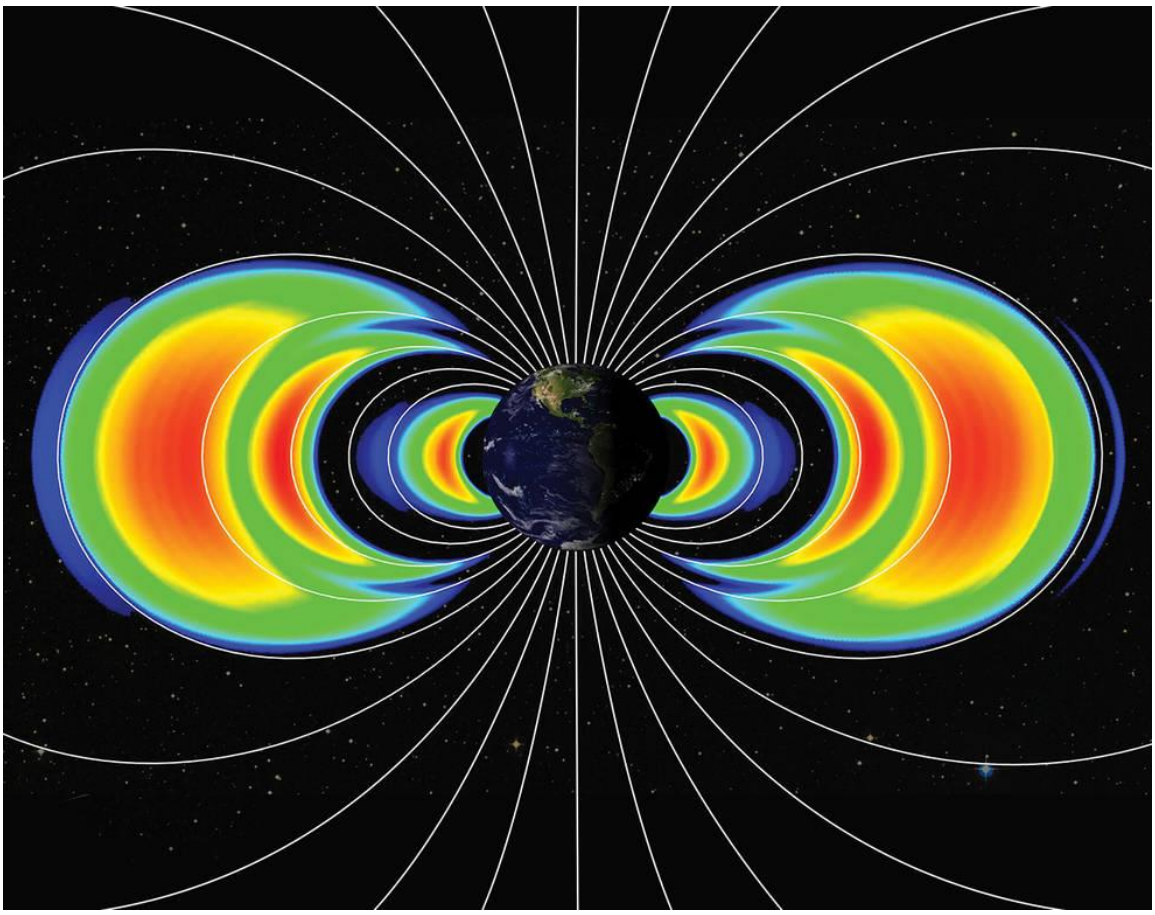

Figura 2.2 – Cinturones de Van Allen [4]

Estos tres tipos de radiación son ionizantes, esto quiere decir que la radiación puede transpasar sustancias y alterarlas. Cuando esto ocurre, esta ioniza los atomos de la sustancia haciendo que pierda sus electrones, además estos electrones liberados pueden interaccionar con los demás átomos de la sustancia y alterarla aún más.

2.2 Efectos de de la radiación en dispositivos electrónicos

Como ya se ha mencionado anteriormente, los entornos de radiación ionizante están conformados por partículas como electrones, protones, neutrones e iones pesados. La radiación puede interactuar de dos maneras [5] con materiales electrónicos:

- Interacción electrónica: La radiación incidente interactúa con la carga circundante del átomo. Las partículas que componen la radiación transfieren energía al átomo, el cual cambia a un nivel más alto su nivel energético y produce la ionización de los electrones del átomo. En materiales electrónicos esto produce que mayor cantidad de electrones estén disponibles para su conducción.

  En el caso del flujo incidente compuesto por electrones, la cantidad de energía transferida tiene pocas probabilidades de ser de un valor tan alto que provoque efectos notables en el dispositivo. Pero en el caso del flujo compuesto por iones pesados es diferente debido a que estos depositan una cantidad de energía superior y la probabilidad de dañar el dispositivo electrónico es mayor debido al aumento de la temperatura generada por el aumento de la conducción de electrones.

- Interacción nuclear: Para que se de este caso la radiación debe interactuar directamente con el nucleo del átomo. Los protones y neutrones de gran energía [6], producidos por los rayos cósmicos galácticos, pueden colisionar con nucleos de materiales electrónicos como el silicio y generar una reacción de espalación. Mucha energía es transferida en esta interacción, y en esta situación el átomo puede ser físicamente desplazado de su localización original o incluso puede fragmentarse liberando

particulas como iones pesados. El impacto de este tipo de interacción puede causar problemas más graves que la interacción electrónica pero la probabilidad de que este suceso ocurra es mucho menor.

Los neutrones térmicos [6][7], neutrones libres con una energía de alrededor de 0,025 eV, deben tenerse en cuenta en estas interacciones debido a que tienen una gran probabilidad de interactuar con ciertos isotopos, como el B^{10}. Cuando un nucleo de B^{10} interactua con un neutrón térmico se produce una reacción que libera una particula Alpha y un Li^{7} que tienen en conjunto una energía aproximada de 2,5 eV que puede ser depositada en un dispositivo electrónico y producir un SEU.

2.2.1 Total Ionizing Dose

Uno de los efectos de las interacciones de la radiación antes descritas es la "Dosis ionizante total" (TID-Total Ionizing Dose), que es producto del cambio de estado energético de electrones de los átomos que interactúan con la radiación que deposita energía en la estructura. En general, toda la electrónica es susceptible a la ionización, pero, si el nivel de interacción es bajo, la carga generada dentro del material semiconductor puede ser rápidamente eliminada sin efectos negativos.

En el caso de la tecnología CMOS es diferente debido a que, como está conformada por una interfase dióxido/silicio, la carga generada dentro del óxido puede quedarse atrapada en la interfase. Esta carga atrapada puede llevar a un incremento de la corriente de fuga o cambiar las características operacionales del dispositivo debido al cambio del potencial de la estructura de la interfase.

Para mitigar estos efectos lo apropiado es proteger los materiales electrónicos con otro material que interactúe con el entorno de radiación para que este pierda energía y que la mayoría de las partículas de la radiación no entren en contacto con la electrónica. El efecto de esta protección no es lineal, como puede observarse en la Figura 2.3, por lo que en caso de estar ante un ambiente radiante en el que el agregado de más protección no cumpla con las condiciones de tolerancia, lo óptimo es reemplazar la protección por una versión más tolerante.

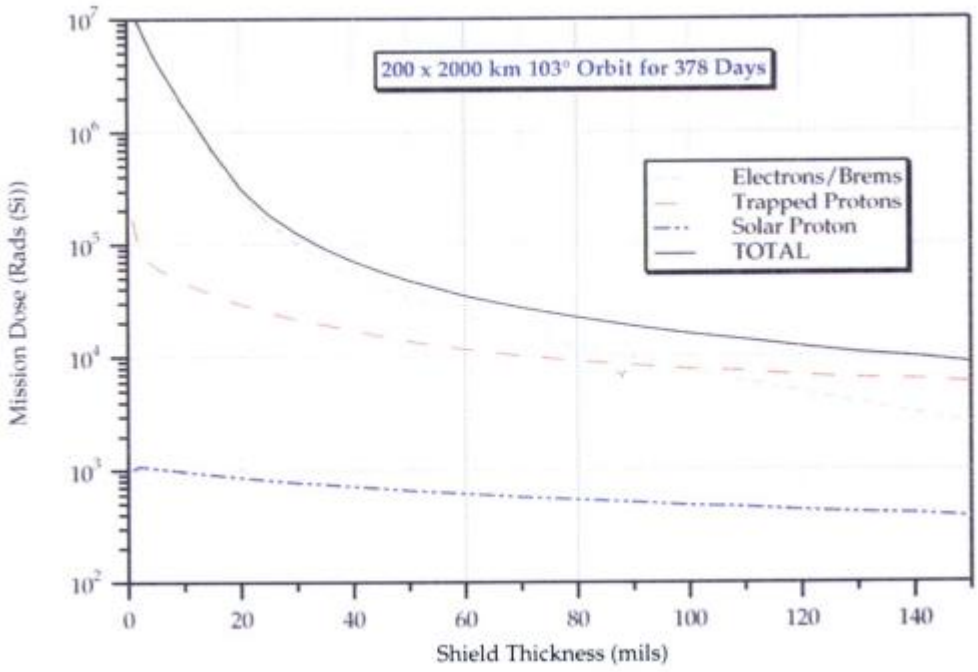

Figura 2.3. Dosis de radiación recibida en función del espesor de la protección [5]

2.2.2 SEE

La radiación cósmica puede afectar de distintas maneras a la electrónica utilizada en misiones espaciales, estos pueden incluirse dentro del fenómeno denominado como Single Event Events(SEE) [8] que está asociado a partículas energéticas individuales (iones pesados y protones) que inciden en zonas sensibles de los circuitos. SEE puede dividirse en distintos tipos de eventos, como, por ejemplo:

- Single Event Upset(SEU): Es un cambio de estado o transitorio inducido por una partícula ionizante como un rayo cósmico o un protón en un dispositivo electrónico. Puede ocurrir en componentes digitales, analógicos y ópticos. En una memoria se manifiesta como la inversión del valor de un bit.
- Single Event Latchup(SEL): Es generado por la inducción de un estado de alta corriente y puede o no causar daños permanentes en el dispositivo, aunque en ambos casos se debe dejar de alimentar el dispositivo para recuperar funcionalidad.
- Single Event Burnout(SEB): Normalmente se refiere a un fallo instantáneo de un transistor de potencia al ser golpeado por una partícula ionizante lo que provoca un cortocircuito, un fallo que puede ser catastrófico para una misión espacial.

Los efectos provocados por la radiación pueden ser a nivel de software o hardware. Los errores provocados en el software no son destructivos en el dispositivo, en cambio, un error en el hardware producto de la radiación puede ser destructivo por lo que un error de este tipo en una misión espacial es inaceptable.

Como medida preventiva, se puede conocer, aproximadamente, la resistencia de los materiales ante los SEE mediante la magnitud denominada Transferencia de Energía Lineal (LET-Linear Energy Transfer) que representa la energía depositada por unidad de longitud cuando una partícula energética atraviesa el material, la unidad de esta medida es $MeV*cm^2/mg$.

Además, se define un umbral para esta magnitud, la cual se denomina Threshold LET (LETth) e indica el minimo valor de LET con el cual una partícula puede alterar el material. Por lo tanto, esta magnitud es importante tenerla en cuenta debido a que es una prueba de si los materiales utilizados serán lo suficientemente resistentes a la radiación como para poder ser utilizados en misiones espaciales sin poner en riesgo a los tripulantes y a la misión.

2.3 Mitigación SEE

Es necesario tomar medidas [9] contra los efectos de la radiación para evitar que esta afecte a una misión espacial. A nivel de software se pueden utilizar distintos métodos, dependiendo de la complejidad del dispositivo, para mitigar los efectos de la radiación.

- CHEQUEO DE REDUNDANCIA CICLICA(CRC): Los bits del dato son transformados en un polinomio. El dato es dividido en modulo-2 por el polinomio generado y el residuo es un carácter CRC que se agrega a la estructura del dato, nuevamente es dividido por el polinomio generado y si el residuo es 0 es que no hay un error en el dato.
- CODIGO DE HAMMING: Detecta la posición de un error simple y la existencia de más de un error en la estructura de un dato. Se basa en ordenar los bits del dato de distintas formas para obtener distintos resultados cuando hay algún error, de esta forma se puede identificar los bits erróneos y corregirlo.
- CODIFICACION CONVOLUCIONAL: Al igual que los anteriores, es un código lineal, pero a diferencia de ellos, no solo se basa en el valor actual del dato sino de los anteriores valores, es decir, tiene una memoria. La codificación, comúnmente, se realiza mediante registros de desplazamiento conectados a sumadores base 2.
- WATCHDOG TIMERS: Cada vez que un dispositivo recibe un mensaje, envía otro para indicar que no ha sufrido ningún error. Si pasado determinado tiempo después de enviar un mensaje a un dispositivo no se recibe ninguna respuesta de este se determina que un “time out” ha ocurrido, lo que indicaría que el dispositivo receptor ha dejado de funcionar o no puede enviar mensajes por lo que se

deben tomar medidas para que vuelva al funcionamiento normal.

- REDUNDANCIA MODULAR TRIPLE: Si se tienen 3 circuitos idénticos y al menos dos de ellos "votan" por una misma salida (un mismo dato) entonces ese dato es correcto. Aunque es un método con buenos resultados y es más rápido que el mayormente utilizado código Hamming, tiene la desventaja de que se requieren 3 veces más electrónica, lo que supone un aumento del espacio y la imposibilidad del uso de dichas compuertas para otras aplicaciones.
- DEPURADO DE CONFIGURACIÓN Y MEMORIA [10]: Se trata de un chequeo periódico de la memoria, detectando alteraciones, para luego usar una configuración parcial dinámica para sobreescribir las alteraciones. Tiene dos responsabilidades principales: detección de errores y su corrección.

  En el caso de memorias dinámicas el depurado es más complejo debido a que sus datos almacenados cambian constantemente. Dos tipos de depuración son por probabilidad o por determinación, depuración probabilística es cuando se depura solo las zonas de la memoria a la que se ha accedido, el problema es que las zonas a las que no se ha accedido pueden ser alteradas y no se depurarían. En la depuración por determinación se realiza un barrido secuencial de todas las zonas de memoria que detecta errores, el problema de este tipo de depuración es que consume muchos recursos.

2.4 Predicción de SEUs

Poder predecir la cantidad de SEUs que pueden darse en una misión espacial es de gran utilidad porque puede servir como soporte de un estudio de la mitigación de SEUs de un determinado dispositivo electrónico planeado para utilizarse en el espacio. En el caso de este trabajo, si como resultado se obtiene que 1 de cada 100 SEUs genera un mal funcionamiento de un dispositivo, ¿puedo decir que es un buen resultado? La respuesta dependerá de la cantidad de SEUs a los que se vera afectado el dispositivo mientras esté operativo.

Los cálculos [11] de la predicción son complejos, pero existen herramientas para realizarlos teniendo en cuenta distintas variables. Los cálculos dependerán del tipo de radiación, para el caso de los iones pesados la ecuación es la siguiente:

$$SEUrate = \int_0^\infty \sigma(\lambda)\mathrm{P}(\lambda)\mathrm{F}(\lambda)\mathrm{d}(\lambda) \quad (1)$$

Donde, λ = LET (Linear Energy Transfer), F(λ) es el flujo diferencial de partículas, P(λ) es un factor de corrección geométrica y σ(λ) es la sección transversal del ion pesado.

Mientras que, en el caso de protones, la ecuación es la siguiente:

$$SEUrate = \int_0^\infty \sigma(\mathrm{E})\mathrm{f}(\mathrm{E})\mathrm{d}(\mathrm{E}) \quad (2)$$

Donde, E es un valor de energía, f(E) es el flujo diferencial de partículas con energía >E y σ(E) es la sección transversal del protón con energía E.

2.4.1 CREME96

Como se mencionó anteriormente, los cálculos para la predicción de SEUs son complejos, pero existen herramientas que permiten realizarlos teniendo en cuenta distintas condiciones. Una de estas herramientas es CREME96 (Cosmic Ray Effects on Micro-Electronics) [12], un software que puede simular el entorno radiante en orbitas alrededor de la tierra y evalúa los efectos de la radiación en sistemas electrónicos que se encuentren en dicho entorno.

Las simulaciones se realizan bajo diferentes condiciones solares para simular el ciclo solar, en el cual la actividad solar varía entre dos extremos de energía emitida. CREME96 posee distintos módulos que permite al usuario poder personalizar el entorno radiante a simular, los parámetros a personalizar son, entre otros: flujos de partículas en órbitas, especificaciones de la órbita, tipo de dispositivo electrónico y protecciones.

3 SPACEWIRE

3.1 Introducción

SpaceWire [13] es un protocolo de comunicación utilizado en redes de dispositivos a bordo de naves espaciales utilizados para el manejo de datos, estos dispositivos pueden ser instrumentos, dispositivos de memoria, dispositivos utilizados para telemetría y otros sistemas a bordo. SpaceWire es simple de implementar y cuenta con ciertas características como: alta velocidad (2 Mbits/s a 200 Mbits/s), bajo consumo, simplicidad, relativo coste de implementación y una arquitectura flexible que lo hace muy útil en misiones espaciales. Además, la transmisión de datos es bi-direccional, full-duplex, lo que permite conectar todo el equipamiento en una misma red.

Dependiendo de la aplicación se pueden utilizar enlaces point-to-point y/o routers. Los routers serán necesarios en el caso de que un dispositivo tenga que enviar o recibir datos de más de un dispositivo, en cuyo caso se tendrán que adaptar los paquetes de datos enviados dado que tendrán que incluir una dirección que indique a que dispositivo se desea enviar dicho paquete.

SpaceWire fue creado a partir de la necesidad de tener una interfaz de comunicación estándar en las misiones espaciales, ya que anteriormente cada fabricante de equipamiento tenía su propia interfaz y protocolo de comunicación por lo que resultaba en un gran coste y un mayor tiempo de integración y testeo de todos los dispositivos funcionando en una misma red.

El protocolo fue publicado en 2003 y desde entonces ha sido adoptado por las agencias espaciales ESA, NASA, JAXA y RosCosmos como un estándar para muchas misiones espaciales y otros usos científicos. Algunas de estas misiones donde SpaceWire es utilizado son: ExoMars rover y BepiColombo.

3.2 Cómo funciona

Los enlaces de SpaceWire son enlaces de datos point-to-point que conectan a un nodo SpaceWire con otro nodo o router. La información a travez del enlace se envía como una secuencia de bits en serie usando dos señales en cada dirección, dichas señales son las conocidas como DATA y STROBE. Estas señales son transmitidas a travez del enlace usando Señalizacion Diferencial de Baja Tension (LVDS-Low Voltage Differential Signalling) que requiere dos cables por cada señal por lo que un cable de SpaceWire contiene cuatro pares de cables.

La sincronización de los bits es conseguida al enviar una señal de reloj a travez del enlace junto con los datos. Esta señal de reloj es generada a partir de las señales DATA y STROBE mediante la operación XOR, se realiza de esta manera con el objetivo de reducir la la desincronización de los datos transmitidos.

3.2.1 Paquetes de SpaceWire

Los paquetes están conformados por tres partes [14]:

- Dirección: En el caso en que se utilice un router que reciba y/o envíe paquetes desde y hacia varios dispositivos, lo primero que se tendrá que enviar será una determinada cantidad de bits que representen la dirección que identifica al dispositivo al que se quiere enviar dicho paquete. Para ello, el router deberá tener almacenada una lista de direcciones de los dispositivos en la red.
- Carga: Esta parte del paquete es el dato en si que se quiere enviar a otro/s nodo/s de la red a través del enlace SpaceWire, está conformada por una cantidad de caracteres de datos determinada por la aplicación y cada carácter está compuesto por: un bit de paridad, un bit de control y 8 bits que corresponden al dato.

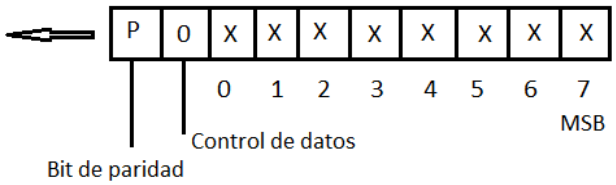

Figura 3.1. Formato de un carácter de datos

El bit "Control de datos" es un bit que indicará si es un carácter de control o un carácter de dato, si es '0' indicará que el caracter es un dato y si es '1' querra decir que el carácter es de control por lo que podrá ser un EOP (End of Packet) o un EEP (Error End of Packet).

- EOP: Es la combinación de bits que va al final del paquete e indica que el paquete se ha transmitido por completo. Como se ha mencionado en el apartado de la carga, el bit de control de datos sirve para indicar y detectar si el carácter es de control o de datos, en el caso de que este sea '0' será un carácter de datos y el receptor estará esperando que se reciba otro carácter de dato o de control, mientras que si es '1' este podría ser el fin de un paquete. Podria ser el fin del paquete, pero, al estar este bit en '1', el MSB del carácter de datos o control funcionará como un bit de control por lo que si este es '0' entonces el carácter será un EOP, en cambio si es '1' indicara que es un EEP.
- EEP: Indica que el paquete ha sido dejado de transmitir prematuramente debido a algún error en el enlace. Cuando ocurra esto el enlace generará un EEP automáticamente y el carácter debería ser descartado. El protocolo intentará recuperarse del error producido en el enlace mediante un reset del mismo y la aplicación debería hacer que se continúe transmitiendo el paquete en el punto en que ocurrio el error o puede solicitar que todo el paquete vuelva a ser enviado.

Figura 3.2. Formato del paquete en SpaceWire

Cabe aclarar que la cantidad de paquetes y la cantidad caracteres de datos que estos contengan no tiene limite por lo que datos de gran tamaño pueden ser transmitidos. La única limitación con respecto a la cantidad de paquetes y caracteres de datos es la capacidad de las memorias disponibles en la aplicación.

3.2.2 Inicialización del enlace

Para poder transmitir datos entre dos dispositivos lo primero que se deberá hacer que establezcan contacto entre ellos, ambos deben saber que el otro está activo y listo para transmitir. La inicialización del enlace consta de varios pasos en los que los nodos se envían unos determinados comandos y se espera una determinada respuesta. Una vez finalizado el proceso de inicialización se procederá a enviar paquetes si los hay disponibles para enviar y si el receptor tiene espacio suficiente para recibirlos.

El proceso puede iniciarse automáticamente después de un reset o puede iniciarse a partir de la orden de la aplicación. Además, durante el proceso pueden ocurrir errores en el enlace que provoquen que el proceso deba ser reiniciado o pueden ocurrir cuando ya haya sido establecido el enlace y que se inicie la recuperación del mismo reseteandolo e iniciando nuevamente la inicialización.

A continuación, se puede observar una versión básica de la maquina de estados que se utiliza para la inicialización del enlace SpaceWire:

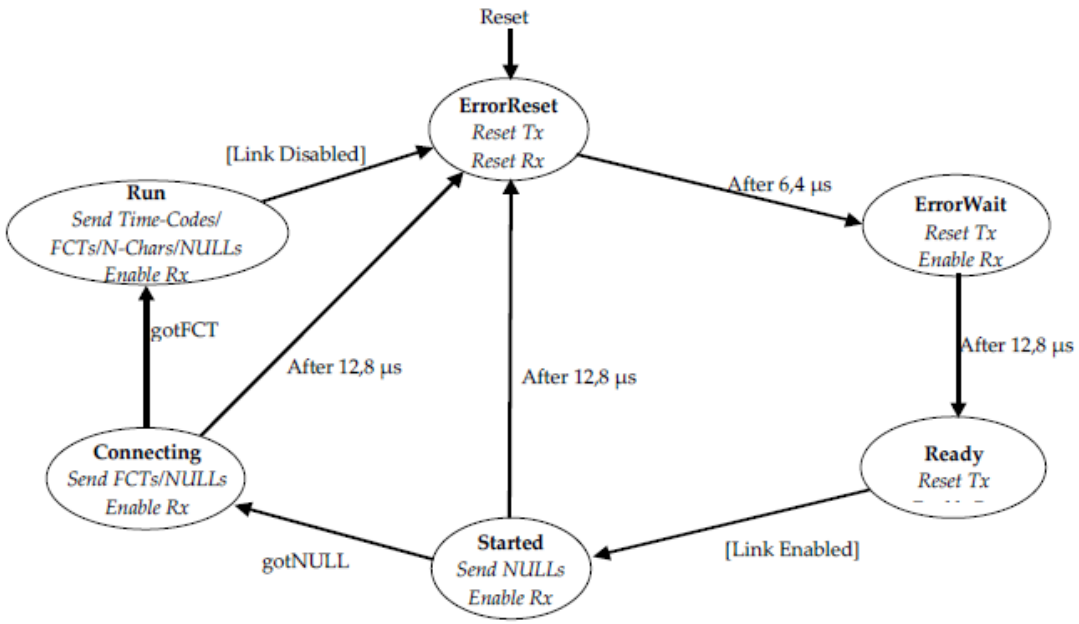

Figura 3.3. Maquina de estados básica de la inicialización del enlace [14].

3.2.3 Errores de enlace

Son varios los errores que pueden producirse en el enlace y que puedan deshabilitarlo temporalmente.

A continuación, se listan dichos errores:

- CreditError: Este tipo de error ocurre cuando datos son recibidos cuando el nodo no esperaba recibir ningún dato. Indica que un error no detectado ha ocurrido en el sistema y por eso se están recibiendo datos cuando no se debería.

- DisconnectError: Este error es generado cuando no se ha mantenido ningún contacto entre dos nodos durante más de 850 ns nominales. El no contacto se define como ninguna variación en los valores de las señales DATA y STROBE en el receptor.

  Para evitar que este error se genere cuando el enlace esté activo y no haya paquetes que enviar, el protocolo envía continuamente un comando denominado NULL para mantener el contacto.

- ParityError: La paridad cubre cada carácter de control y cada carácter de control en varios niveles. Se detectará un error de paridad cuando el número de errores sea impar.

- EscapeError: El comando NULL esta formado por dos comandos: ESC + FCT. Si se da el caso que se envía ESC seguido de otro comando de control, se estaría enviando otro comando distinto de NULL y esto es un error.

Estos errores detendrán la transmisión de paquetes, si se estaba ocurriendo, y en ese caso se generará un EEP, en caso contrario no se generará un EEP, pero en ambos casos se iniciará la recuperación del enlace.

3.3 Misiones espaciales

3.3.1 ESA ExoMars

Es una misión [15] de ESA que incorpora un rover. El rover lleva consigo un conjunto de instrumentos científicos dedicados a la investigación exobiológica y geológica de Marte. Viajará a travez de Marte en busca de signos de vida en el pasado y en el presente, recolectando y analizando muestras de rocas en la superficie y en la subsuperficie, a una profundidad de 2 m.

Ademas de instrumentos para realizar investigaciones científicas, cuenta con cámaras con visión panorámica que serán de utilidad para propósitos de navegación.

SpaceWire será utilizado para transmitir las imágenes captadas por las camaras hacia una unidad de procesamiento de imágenes. Además, un router SpaceWire es utilizado para interconectar varios dispositivos que utilizan este protocolo para comunicarse.

El inicio de la misión dará lugar en 2020.

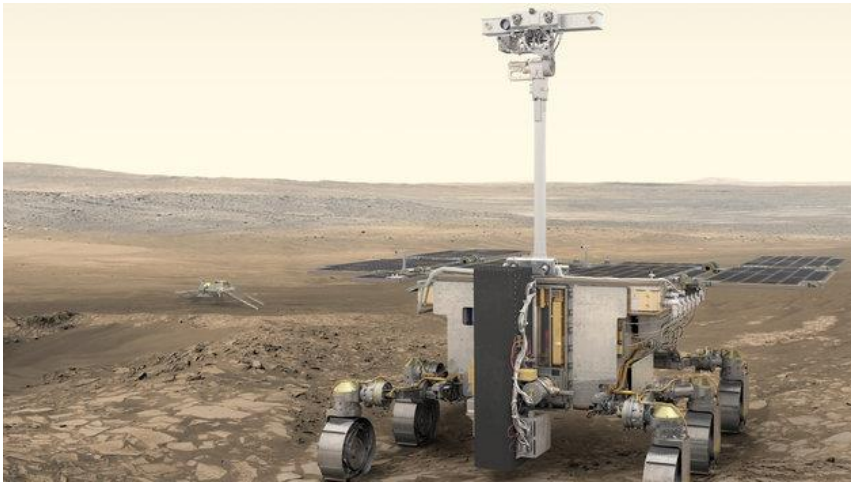

Figura 3.4. ExoMars Rover [16]

3.3.2 BepiColombo

Es una misión [17][18] en cooperación entre ESA y JAXA que tiene como objetivo explorar Mercurio. Es un gran reto debido a la cercanía al Sol, que dificulta a la nave llegar al destino y sobrevivir en ese entorno.

Está compuesta por tres modulos:

- Orbitador Planetario de Mercurio (MPO- Mercury Planetary Orbiter): Diseñado por ESA, es un modulo científico caracterizado por una serie de instrumentos de observación para estudiar la superficie de Mercurio y su campo gravitatorio.
- Orbitador Magnetosférico de Mercurio (MMO – Mercury Magnetospheric Orbiter): Diseñado por JAXA, es un modulo científico estabilizado por rotación construido con el objetivo de estudiar el campo magnético de Mercurio.
- Módulo de Transferencia de Mercurio (MTM – Mercury Transfer Module): Diseñado por ESA, este módulo es el que cargara con los otros dos módulos.

El módulo MPO utiliza SpaceWire para el manejo de datos recogidos. Los instrumentos de medida están conectados a la memoria de almacenamiento mediante enlaces SpaceWire y tres routers SpaceWire están integrados en la memoria.

El módulo MMO también utiliza SpaceWire para configurar, controlar, y recolectar datos de todos sus instrumentos. Debido a la poca capacidad de espacio y el bajo consumo, una versión especial de SpaceWire ha sido instalada en este modulo, esta versión inicializarse y transmitir a 2 Mbits/s.

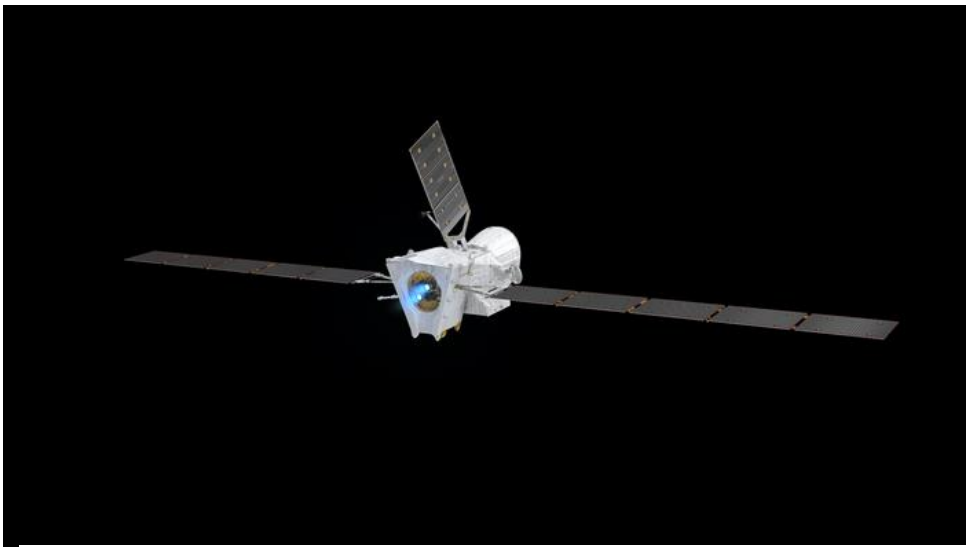

Figura 3.5. Modulo BepiColombo [19]

3.4 Otros tipos de comunicación en el espacio

Las naves, tripuladas y no tripuladas, que son enviadas a explorar el espacio exterior tienen una gran cantidad de dispositivos electrónicos y son altamente dependiente de estos ya que un fallo en la electrónica puede resultar en una catástrofe. Por ello, la adaptación de la electrónica para operar en un entorno espacial repleto de radiación y otros fenómenos que pueden afectar la misión de distintas maneras es vital.

Todos estos dispositivos electrónicos están, en mayor o menor medida, interconectados entre si para cumplir con las necesidades de la misión por lo que es de gran importancia que se tenga una comunicación fiable entre ellos. Además de la comunicación entre los dispositivos electrónicos dentro de una nave espacial, es importante tener una adecuada comunicación tierra-espacio para la transmisión de datos obtenidos y poder mantener contacto con los humanos en el espacio. Por ello es importante que se implementen protocolos de comunicación capaces de resistir, en cierta medida, los efectos de la radiación y así no depender exclusivamente de una protección a nivel de hardware. Además, codigos de corrección de canal son utilizados para corregir errores en la transmisión de datos en canales.

A continuación, se comentarán algunos protocolos de comunicación y códigos de corrección de canal utilizados en el espacio-espacio y tierra-espacio:

- LDPC (Low Density Parity Check) [20]: Es un codigo corrector de errores linear, un método para transmitir datos a travez de un canal ruidoso mediante el uso de algoritmos iterativos. El avance de la tecnología ha permitido la implementación y el avance de los mismos, de tal manera que el ratio de transmisión de datos a travez de un canal puede alcanzar valores muy altos y por ello cada vez es utilizado en más aplicaciones. La NASA lo utiliza en la transmisión por radio frecuencia donde la cantidad de datos a transmitir es muy grande y los errores en el canal serían un gran problema sin este código.
- PROXIMITY-1[21]: Protocolo de comunicación diseñado únicamente para transmitir datos entre sondas, modulos de aterrizaje, rovers, constelaciones en orbita[2] y satélites de retransmisión en un entorno próximo. Está diseñado para transmitir entre 1 y 100.000km, es bidireccional y transmite datos mediante modulación FSK y PSK.

3.4.1 SoCWire

Las misiones espaciales cada vez tienen objetivos más grandes y, por consiguiente, mejoras en la tecnología son necesarias. Detectores de alta resolución con alta transferencia de datos y gran volumen de datos requieren un intenso procesamiento de datos a bordo debido a que el canal de transferencia de datos es pequeño. Una gran mejora sería, la reconfigurabilidad en vuelo y la existencia de módulos dinámicos reconfigurables para fines de mantenimiento y funcionalidad adaptativa en ejecución.

Para conseguir estas mejoras se ha desarrollado una arquitectura NoC (Network-on-Chip) llamada System-on-Chip Wire (SoCWire) [22].

Esta arquitectura está basada en SpaceWire y tiene 6 módulos reconfigurables que pueden ser operados por un router. Se han hecho modificaciones en SpaceWire, los caracteres se transmiten en paralelo en vez de en serie, lo que da una gran mejora a la velocidad de transferencia y se ha implementado un escalado del tamaño de la palabra a transmitir para permitir mayores ratios de transferencia de datos. Además, se podrían reducir los tiempos de inicialización del enlace de 19,2 us a 400 ns y el tiempo de recuperación del enlace de 20,05 us a 1,13 us.

El router permite la transferencia de paquetes que llegan a un nodo hacia otro nodo. Este router y el protocolo están basados en el estándar SpaceWire y soporta palabras de largo 8-128 y 2-32 puertos con una interfaz entrada-salida totalmente simétrica.

4 FT-UNSHADES2

FT-UNSHADES [23] (Fault Tolerance – University of Sevilla Hardware Debugging System) es una plataforma que permite la simulación de los efectos de un ambiente de radiación ionizante en un circuito integrado.

La plataforma está compuesta por una FPGA de control conectada a una memoria RAM y a un servidor para almacenamiento y transmisión de datos. A su vez, esta tarjeta está conectada mediante puertos PCI-Express a dos tarjetas, donde cada una de ellas esta compuesta por dos FPGAs, una FPGA de servicio y otra FPGA de testeo. La tarjeta de servicio configura, realiza configuraciones parciales para la ejecución de las inyecciones de errores, transmite las entradas del diseño y recoge los datos devueltos de la FPGA de testeo.

Todas las FPGA son Virtex XC5VFX70T debido a la facilidad de hacer cambios en su configuración, esto permite simular el mismo efecto que los SEUs provocados por la radiación ionizante generan en dispositivos electrónicos, ya que lo que hacen es cambiar el valor de bits de registros y configuración en puntos sensibles del circuito. Para simular el efecto de los SEUs se podrán realizar modificaciones en bits de registros del diseño, a esto se le llama "inyección de fallos" debido a que se está generando un SEU en determinado punto del diseño, las inyecciones se pueden realizar de distintas maneras, como se explicará en el siguiente apartado.

Como ya se ha mencionado, la plataforma ejecuta dos versiones del diseño: GOLDEN y SEU. GOLDEN es la versión del diseño en la que no se inyectan fallos, por lo que su funcionamiento representaría un ambiente donde no hay radiación ionizante o no se produce ningún SEU, mientras que SEU es la versión del diseño en que los fallos son inyectados y simula al diseño en un ambiente con radiación ionizante y la posibilidad de verse afectado por SEUs. Las dos versiones son comparadas para poder detectar posibles errores de funcionamiento generados por los SEU inyectados.

Puede accederse a la plataforma via web y está conectada a un servidor donde almacena proyectos creados por los usuarios por lo que puede ser utilizada en todo momento y desde cualquier ubicación. Se debe tener en cuenta que los diseños de circuitos deben ser sintetizados para la FPGA utilizada por la plataforma, Virtex XC5VFX70T. Se han desarrollado dos guias: UFF 3.5 User Guide y UFF 3.5 Getting Started Guide, para el correcto uso de la plataforma y el conocimiento de todas las funcionalidades de esta.

FTU2 es una herramienta muy potente debido a que permite a los usuarios diseñar un circuito electrónico teniendo en cuenta lo resistente que puede ser ante la radiación ionizante. Implementando el diseño en la plataforma y luego de la correspondiente inyección de fallos, esta dará resultados sobre el efecto de SEUs en el diseño por lo que el usuario tendrá la oportunidad de modificar, a partir de los resultados, el diseño para disminuir la sensibilidad contra la radiación ionizante en las correspondientes zonas del circuito.

La posibilidad de simular un ambiente de radiación ionizante también supone una reducción de costes, ya que, a nivel de software hay multiples posibles modificaciones del diseño de forma que la posibilidad de un fallo en el funcionamiento provocado por un SEU sea reducida al punto de que no sea necesaria la inversión en una mejora del hardware o los materiales utilizados para la reducción de los efectos de la radiación.

4.1 Modos de trabajo

A disponibilidad del usuario y en función de su objetivo, existen dos modos de trabajo en la plataforma.

4.1.1 Modo DEBUG

En este modo el usuario podrá realizar inyecciones de fallos de forma manual en el registro o señal que desee, habilitando, previamente, dicho registro o señal para su inyección. Además, podrá modificar sus valores en el ciclo que crea oportuno. En este modo se podrán observar las señales a inyectar fallos y los pines de salida definidos en el diseño, es muy útil este modo debido a que podremos ver en que pin de salida o señal y en que ciclo se generan errores producto de una previa inyección de fallos.

Es de gran utilidad debido a la facilidad y rapidez con la que se pueden detectar fallos, y comprobar si el diseño funciona correctamente una vez inplementado en la FPGA. Obviamente no se espera que los usuarios obtengan resultados sobre la resistencia a la radiación ionizante de sus diseños mediante la inyección manual en cada señal y ciclo posible dado que conllevaría demasiado esfuerzo y tiempo, por ello existe el modo campaña que a continuación se explicará.

4.1.2 Modo CAMPAÑA

Este modo permite al usuario automatizar el proceso de inyecciones de fallos de manera que de varias maneras posibles y con distintas configuraciones posibles se puedan realizar la cantidad de inyecciones que se desee.

Una campaña es la realización de un determinado número de ejecuciones del diseño realizado en determinado rango de ciclos en donde en cada ejecución se puede inyectar uno o más de un fallo en determinados registros. Los tipos de campañas que pueden realizarse son:

- Campaña aleatoria: En este caso se inyectarán fallos de manera aleatoria en el rango de ciclos y registros establecidos. El usuario será capaz de configurar parámetros como el rango de ciclos, registros a inyectar y errores inyectados por ejecución, de tal manera que el usuario pueda personalizar la campaña en función de sus necesidades.
- Campaña exhaustiva: Se inyectará en cada registro y cada ciclo que se haya definido, dados parámetros son los únicos que el usuario tendrá que definir. Este tipo de campaña es la que entregará los mejores resultados ya que cubrirá todos los casos posibles de errores de funcionalidad provocados por la radiación ionizante. Lo negativo de esta campaña es que debido a que, dependiendo del diseño, la cantidad de ejecuciones a realizar podría alcanzar valores tan altos que pueda hacer que la plataforma no soporte tal proceso. Por ello, una campaña aleatoria es la mejor opción en caso de que el diseño sea considerablemente grande ya que con una cantidad de ejecuciones proporcional a la cantidad de ciclos y registros sin que llegue a un valor muy alto, se pueden obtener resultados en los que se puede confiar y tomar desiciones sobre el diseño.
- Inyector de archivos: Se realiza una campaña de inyecciones a partir de un archivo previamente creado por el usuario. Esta campaña permite una mayor personalización de las inyecciones, comparadas con la aleatoria y exhaustiva. El formato de dicho archivo es explicado en el manual de FTU2.
- Campaña personalizada: Es una alternativa al tipo de campaña anterior y se diferencia en que las inyecciones son realizadas desde una función TCL.

4.2 Obtención de resultados

Luego de realizar la campaña que el usuario crea conveniente, FTU2 provee varios ficheros con resultados e información sobre la campaña. Estos son:

- run.tcl: Es un script que replica la campaña ejecutada.
- status.txt: Muestra estadísticas sobre la campaña durante su ejecución. Algunas de estas son, entre otras: tiempo que ha tardado en realizarse y ejecuciones realizadas por segundo.
- reg_names.txt: Es una lista de todos los posibles registros en los que la campaña, dependiendo del tipo, ha podido inyectar fallos.
- injections.csv: Describe las inyecciones realizadas en la campaña. Indica el ciclo, seguido del registro en que ha sido realizada la inyección del fallo en una ejecución.

```
3258:219;
3515:125;
3547:303;
3521:276;
3922:568;
3121:81;
3472:575;
```

Figura 4.1. Formato de inyecciones

- damages.csv: En este fichero se muestran los resultados más importantes de la campaña. Se puede observar el ciclo en que un error en los pines de salida definidos ha sido detectado. La plataforma tiene la opción de que en este fichero se muestre el resultado XOR entre los pines de salida de la versión GOLDEN y la versión SEU para así poder saber en que pines ha sido detectado el error.

```
;
3444:004000;
;
3417:001000;
3946:000080;
3424:004000;
```

Figura 4.2. Resultados XOR de los daños detectados

5 PREPARACIÓN Y ELECCIÓN DEL DISEÑO

5.1 Diseño del enlace

Debido a que el objetivo del proyecto es analizar la respuesta del protocolo SpaceWire ante los efectos de la radiación cósmica durante misiones espaciales, no es necesario utilizar un diseño muy complejo el cual podría ser utilizado en una aplicación que requiera más de dos dispositivos conectados a uno o más routers. Pero, dado que todos los dispositivos trabajarían con SpaceWire, los efectos de la radiación afectarán de la misma manera a todos los dispositivos por lo que estudiando el caso más simple de interconexión entre nodos SpaceWire se podrán obtener resultados que den conclusiones sobre la robustez de SpaceWire ante este tipo de radiación. Estas conclusiones ayudarán a mejorar, si es necesario, la mitigación de la radiación en toda la red SpaceWire.

El diseño elegido para el estudio es un nodo emisor y un nodo receptor, en donde solo el emisor envía datos a travez del enlace. A continuación, se muestra el diagrama de bloques del diseño realizado en VHDL para el proyecto:

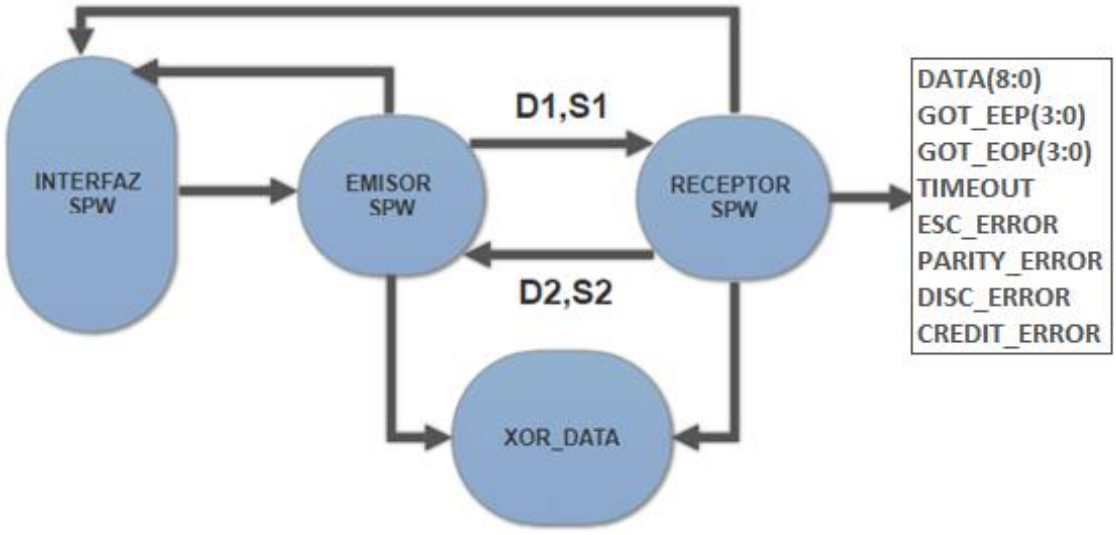

Figura 5.1. Diagrama de bloques del diseño utilizado para el proyecto

Funcionamiento de los bloques:

- EMISOR Y RECEPTOR SPW: Estos bloques han sido proporcionados por ESA con fines académicos únicamente, por ello durante todo el proyecto se tratarán como una caja de la cual solo se utilizarán sus pines de entrada y salida. Ambos bloques son internamente iguales, la única variación es la conexión de sus pines de entrada y salida que han sido utilizados de manera que puedan tratarse propiamente como emisor y receptor.
- INTERFAZ SPW: Es un bloque que funciona como una interfaz de la aplicación que utiliza la red SpaceWire, se encarga de tomar desiciones sobre las acciones a realizar. Se ocupa de suministrar los datos a transmitir por el emisor y de habilitar el mismo para que realice tal operación. Además, en caso de que ocurra un error en el enlace y se genere un EEP, esta indicará que se vuelva a enviar el paquete o no. Tambien, dependiendo del espacio en las memorias, continuará suministrando paquetes o no.
- XOR_DATA: Es un bloque auxiliar que contiene dos memorias que son un reflejo de las memorias utilizadas por el emisor y receptor SpaceWire. Contienen los mismos datos y estos son cargados al mismo tiempo que las del emisor y receptor. Una memoria contiene los datos a emitir y la otra contiene los datos recibidos por el receptor, el objetivo es comparar los datos almacenados por ellas y detectar si se ha producido algún error en los datos transmitidos y/o si se genera un EEP.

Hay que aclarar que los bloques INTERFAZ_SPW y XOR_DATA son únicamente utilizados para facilitar la detección de errores, estos fueron implementados en la plataforma FTU2 pero no se han inyectado fallos en sus registros y señales dado que estos no pertenecen al protocolo en si.

5.2 Datos a transmitir

El formato que debe tener el dato a transmitir es:

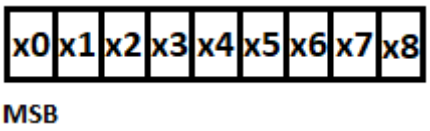

Figura 5.2. Formato de carácter de dato a transmitir

Sienbo el MSB el bit de control que indicara si es el final de paquete, siendo erróneo o no.

El bit x8(msb) indicará si el paquete es un EEP o un EOP en caso de que x0 sea ‘1’. Al suministrar los datos que la aplicación solicita transmitir, entre estos datos no debería haber ningún EEP dado que estos no deberían ser almacenados por las memorias. En caso de que un EEP sea almacenado, al ser recibido por el receptor este dejará de transmitir los caracteres que le seguían a este y este, obviamente no es un suceso deseado.

Los datos elegidos para transmitir no tienen ninguna partícularidad ya que lo que se pretende es simular el proceso de transmisión de datos que podría darse en cualquier aplicación que utilice SpaceWire para ver como se comporta el protocolo ante la inyección de SEUs en estos datos de memoria y en los demás registros y señales.

Por lo tanto, los datos elegidos para transmitir son una secuencia de caracteres de datos que van desde el 0 al 15 en decimal finalizando con un carácter de control EOP para indicar al emisor y receptor que el paquete acaba en ese punto y debe estar a la espera de un siguiente paquete.

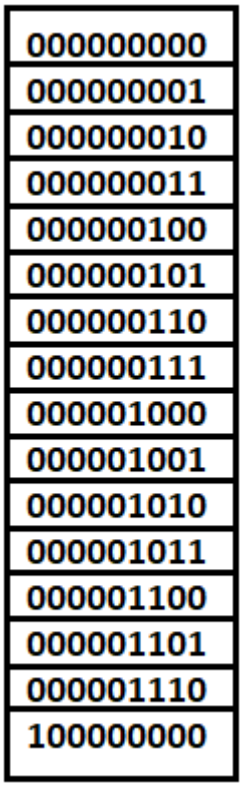

Figura 5.3. Secuencia de datos a transmitir

Como puede observarse, la cantidad de caracteres es 16. Esta cantidad podría haber sido una muy superior, pero, dado que un aumento del tamaño haría al diseño más pesado y la cantidad de ciclos que duraría el proceso de transmisión seria mayor, solo haría que la campaña de inyección en FTU2 durara más tiempo y consumiera más recursos. Además, dado que la campaña de inyección es aleatoria, un aumento de ciclos que se correspondieran al periodo en que se produce la transmisión provocaría un aumento de la probabilidad de que las inyecciones ocurran durante este periodo y no en otro ciclo en el que el protocolo se encuentra en otro estado y que un SEU en el pudiera generar un error que aportara más resultados.

5.3 Interpretación de salidas del diseño

Los 22 pines de salida del diseño son los que he creido conveniente implementar de tal manera que la mayor cantidad de errores posibles fueran detectados con la mayor facilidad posible a partir de los resultados entregados por FTU2.

A continuación, se detalla brevemente lo que representa cada pin:

- DATA: Es un carácter de 9 bits que representa el valor leído por la memoria del receptor. Es decir, es una salida que muestra los caracteres de datos y control recibidos por el receptor, esta va cambiando debido a que a medida que muestra el ultimo carácter recibido hasta que otro sea recibido. A partir de este y los resultados de las inyecciones en FTU2 se podrá ver si se ha producido algún error en los datos transferidos, si se ha producido un EEP inesperadamente, u otros errores que serán detallados en el CAPITULO 6.
- GOT_EEP: Proviene del bloque XOR_DATA e indica si el comando de finalización del paquete es un EEP. Es útil para saber si se ha producido una modificación en los datos transmitidos generando que el receptor lea un EEP sin que haya ocurrido un error en el enlace, o que ocurra un error en el enlace y no se genere un EEP como se debería.
- GOT_EOP: Proviene del bloque XOR_DATA e indica si el comando de finalización del paquete es un EOP. Sirve como verificación de que el paquete ha sido transferido por completo.
- TIMEOUT: Indica que la finalización de la transferencia de un paquete ha superado un valor máximo de tiempo (1 us).
- ESC_ERROR, PARITY_ERROR, DISC_ERROR, CREDIT_ERROR: Estos 4 pines representan 4 posibles errores que pueden darse en el enlace y que detendrían la transmisión del paquete que se este transmitiendo para luego generar un EEP indicándole a la interfaz externa que se ha producido tal error.

6 RESULTADOS DE INYECCIONES

En este capitulo se expondrán los resultados obtenidos de la campaña de inyecciones de fallos realizada mediante FTU2 sobre el enlace SpaceWire diseñado en VHDL.

Como se ha explicado en el capitulo 4, la plataforma FTU2 tiene distintos modos de trabajo y entre estos el modo campaña permite la realización de distintos tipos de campañas. El primer paso para obtener los resultados fue implementar el diseño realizado en la plataforma y comprobar que el funcionamiento era correcto mediante el uso del MODO DEBUG, una vez finalizado este paso se continuo con la realización de la campaña.

La comprobación del funcionamiento puede comprobarse en la siguiente imagen:

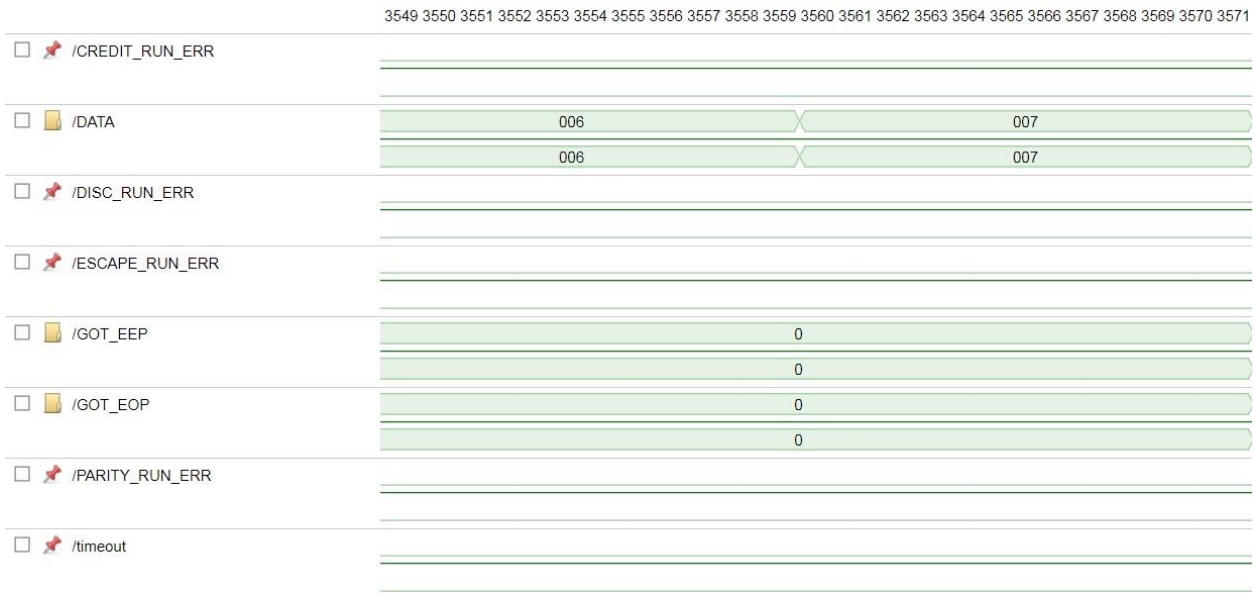

Figura 6.1. Funcionamiento del diseño en modo debug

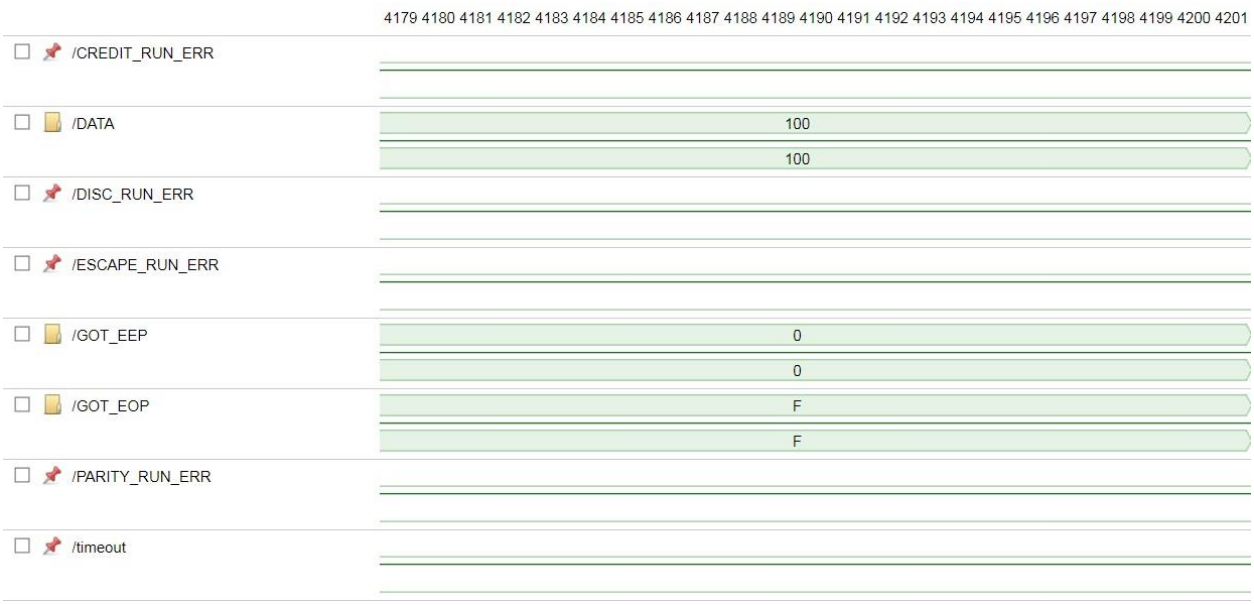

Figura 6.2. Funcionamiento del diseño en modo debug

En las figuras 6.1 y 6.2 puede observarse el funcionamiento del diseño en modo debug. Como medida para verificar que este funciona correctamente puede utilizarse el valor de DATA, que representa las palabras que conforman el paquete recibido por el receptor. Debido a que esta es la ultima etapa del diseño, sabiendo que los datos transmitidos son correctos, puede verificarse el correcto funcionamiento del diseño implementado en FTU2.

Aún así, se ha comprobado el correcto funcionamiento de señales internas para obtener unos datos lo más fiables posibles.

6.1 Campaña de inyecciones

Debido a que el protocolo SpaceWire se podría decir que está dividido en dos etapas, la inicialización del enlace y la transmisión de paquetes, he decidido realizar dos campañas de inyecciones de fallos diferentes en estas dos etapas. La primera campaña fue realizada en el rango de ciclos [100,3000] debido a que este rango cubre todo el proceso de inicialización del enlace, creo que inyectar fallos durante este proceso es importante debido a que es la base para que distintos dispositivos puedan entablar una comunicación y, además, como la inicialización es una etapa que dura aproximadamente 25 us, en los que si ocurre algún error en el enlace provocará que el mismo se reinicie y se tenga que repetir el proceso de inicialización lo que puede retrasar considerablemente el funcionamiento de la red SpaceWire. La segunda campaña se realizó en el rango de ciclos [3200, 3970] que es cuando el enlace esta inicializado y se realiza la transmisión de los paquetes disponibles para enviar.

Teniendo en cuenta la cantidad de ciclos que duran estas etapas y que la cantidad de bits en los que inyectar alcanza valores muy altos en ambos casos, he optado por realizar dos campañas aleatorias con, para el primer caso, 10000 ejecuciones, mientras que, para el segundo caso, 50000 ejecuciones, ambas con un fallo en cada una y 10 daños registrados. Como se ha comentado anteriormente, lo optimo sería realizar una campaña exhaustiva que cubra todos los posibles casos de error, pero debido a los recursos que consumiría y el tiempo que llevaría, no es factible por lo que una campaña aleatoria con la cantidad de ejecuciones mencionadas se adapta a los objetivos del trabajo y los recursos que se tienen. En la primera campaña la cantidad de ejecuciones es bastante inferior que la segunda campaña debido a que creo que no es tan importante como la segunda ya que un error en esa etapa no supondría un gran problema ya que no se estarían transmitiendo datos relevantes para la aplicación llevada a cabo, por ello un estudio más simple es realizado en el caso de esta etapa.

6.2 Clasificación de errores

Una vez obtenidos los resultados de las inyecciones realizadas en FTU2, se ha procedido a tratarlos utilizando Matlab. Para ello se ha creado un script que lee los datos almacenados en el fichero damages.csv generado en la plataforma y dependiendo del valor del resultado XOR se identificarán los errores producidos para crear, a partir de ellos, una clasificación de los errores.

Como se ha mencionado en el anterior apartado, se ha configurado la campaña para que en los resultados se almacenen 10 registros de daños a partir de la inyección del fallo. Para tratarlos los resultados se han creado tablas en Excel que almacenan los 10 daños registrados para posteriormente guardarlas en un archivo .txt capaz de leerlo con Matlab y poder automatizar la generación de la clasificación de fallos mediante un script.

A continuación, se presenta el diagrama de flujo seguido en el script creado en Matlab para generar la clasificación de fallos.

6.2.1 Diagrama de flujo de procesado en Matlab

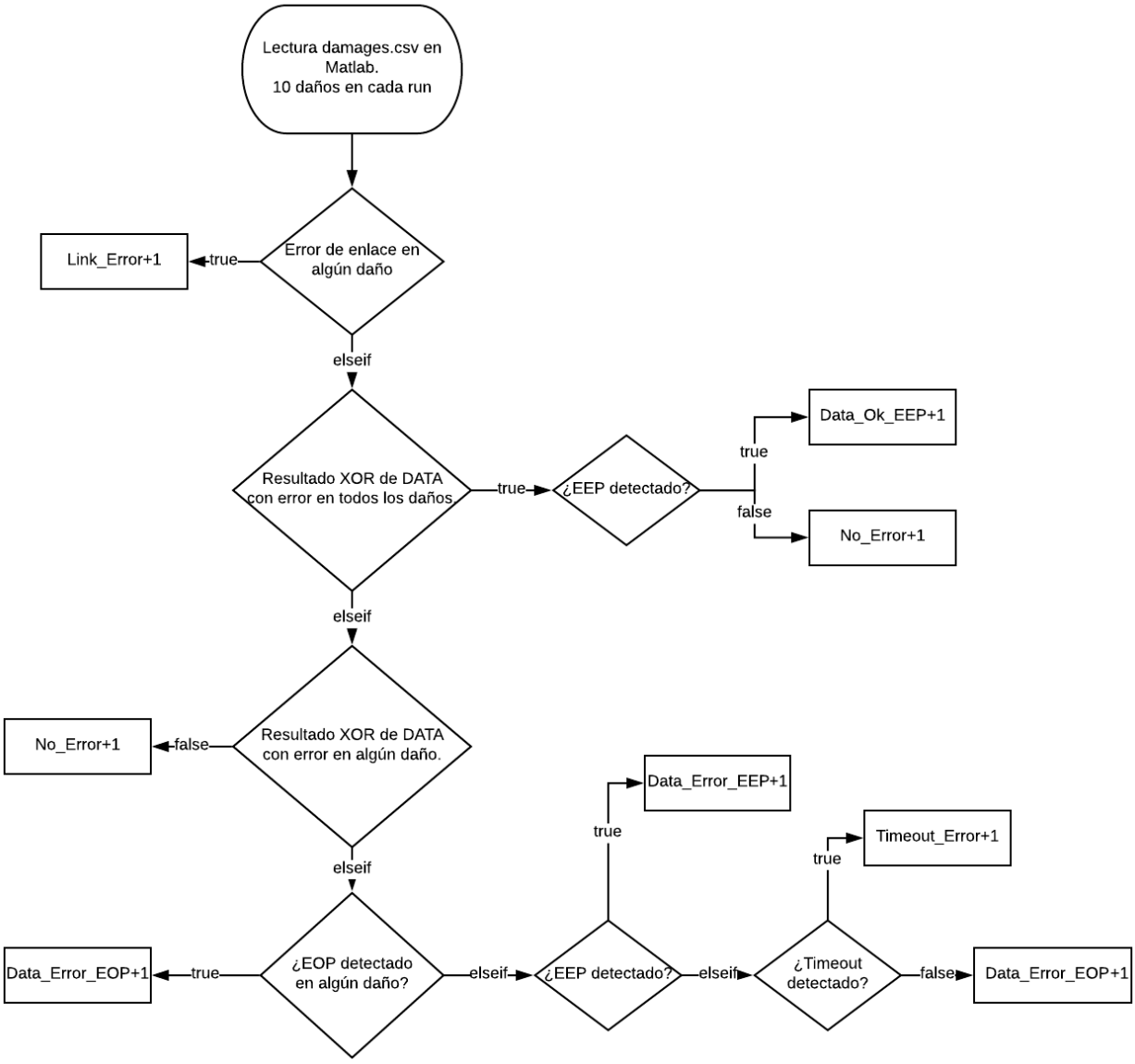

Figura 6.3. Diagrama de flujo para la clasificación de fallos

A continuación, se describen los errores posibles a generarse debido a las inyecciones de fallos:

- LINK_ERROR: Se produce un error en el enlace que provoca que este trate de recuperarse reiniciándolo para volver a inicializarse. En el caso de que se produzca durante la transmisión de un paquete, este finalizará prematuramente y generará un EEP.
- DATA_ERROR_EOP: El paquete se transfiere por completo, finalizándolo con un EOP, pero los datos transmitidos son incorrectos.
- DATA_OK_EEP: No hay errores en los datos transmitidos y el paquete es recibido por completo en el receptor, pero en vez de un EOP al final del paquete, se genera un EEP. Esto provoca que el receptor lo tome como no finalizado completamente debido a que el receptor no sabe el tamaño del paquete.
- DATA_ERROR_EEP: El fallo inyectado modifica el dato para convertirlo en un EEP o se ha generado un error en el enlace que finaliza prematuramente el paquete generando un EEP.
- TIMEOUT_ERROR: El paquete supera un tiempo máximo (1 us) para ser transferido por completo.

Cabe aclarar que los errores de enlace pueden ser producidos por cuatro errores detallados en el capitulo 3, estos errores son: CreditError, ParityError, DiscError y EscError. Debido a que el efecto en el enlace es el mismo, en el script han sido detectados como un único error.

Concluida la descripción de los errores, se procede a clasificar los errores generados en las dos etapas:

ETAPA DE INICIALIZACIÓN DEL ENLACE:

CLASIFICACIÓN DE FALLOS	**RESULTADOS**	**PORCENTAJE**
Inyecciones totales	10000	100%
LINK_ERROR	306	3%
DATA_OK_EEP	0	0%
DATA_ERROR_EEP	10	0.1%
DATA_ERROR_EOP	1425	14%
TIMEOUT_ERROR	7	0.1%
NO_ERROR	8252	82.8%

Tabla 6.1. Clasificación de datos durante la inicialización del enlace

ETAPA DE TRANSMISIÓN:

CLASIFICACIÓN DE FALLOS	RESULTADOS	PORCENTAJE
Inyecciones totales	50000	100%
LINK_ERROR	1173	2.5%
DATA_OK_EEP	0	0%
DATA_ERROR_EEP	408	1%
DATA_ERROR_EOP	9349	19%
TIMEOUT_ERROR	77	0.15%
NO_ERROR	38993	77.35%

Tabla 6.2. Clasificación de fallos durante la transmisión de datos

En la etapa de inicialización, según los resultados obtenidos de la campaña realizada en FTU2, se puede ver que las inyecciones no generan una gran cantidad de errores y esto se debe a que la mayor parte de este periodo es un estado de espera en el cual parte de los bloques no están en funcionamiento por lo que la mayoría de las inyecciones no les afectan.

A partir de la campaña de inyecciones realizada en la etapa de transmisión de datos puede observarse que el error que más veces se produce, con 19%, es un error en los datos transmitidos no detectado por el bit de paridad. El paquete es transmitido por completo con errores en las palabras que lo conforman, la gravedad de este error dependerá del tipo de datos que se estén transmitiendo. Pueden aplicarse distintos tipos de métodos para lograr corregir estos errores en los datos transmitidos.

El segundo error que más veces ha tenido efecto, con un 2.5%, es el error en el enlace que finaliza la transmisión de los paquetes prematuramente para que posteriormente el protocolo intente recuperarse reiniciando el enlace. Es un error que puede causar problemas debido a que el paquete no es transferido por completo y depende de la aplicación para que se vuelva a transmitir, además, durante el proceso de reinicialización el enlace esta suspendido y a el pueden llegar, desde otros nodos, nuevos paquetes para enviar sin la posibilidad de hacerlo y generando un retraso en el sistema. Algo a tener en cuenta es que los nuevos paquetes que se almacenan en la FIFO para ser enviados no se pierden, sino que la FIFO indica que está llena y que no puede recibir más paquetes por lo que hasta que el enlace no esté nuevamente activo no se perderá ningún paquete.

Con 0.15% y 1%, los paquetes que no llegan a transferirse antes de un determinado tiempo y los errores que transforman una palabra en un EEP parecen ocurrir con muy poca frecuencia. El primero, en caso de que se produzca, podría generar un retraso en la red y este retraso puede ser producto de un error que acabe transformándose en un error del enlace provocando de esta manera la reinicialización del mismo como medida de recuperación, mientras que el segundo error, no crea ningún problema en el enlace, sino que el paquete es finalizado prematuramente y la aplicación se debería encargar de recuperar dicho paquete.

6.3 Conclusión

Este capítulo presenta un diseño realizado en VHDL de un enlace emisor-receptor simulando la conexión entre dos dispositivos electrónicos dentro de una red que utiliza el protocolo SpaceWire para la transmisión de datos entre dichos dispositivos, los cuales son utilizados en una misión espacial. A continuación, el diseño se implementó en la plataforma FTU2 para realizar una campaña de inyecciones de fallos que simulará un entorno de radiación ionizante en el que estos dispositivos electrónicos se pueden encontrar durante una misión espacial.

Una vez obtenidos los resultados de la campaña de inyecciones, se procedió a tratarlos mediante un script en Matlab con el propósito de crear una clasificación de los errores producidos en el diseño por los fallos

inyectados para así conocer como es la mitigación de SEUs de SpaceWire en estos entornos de radiación ionizante mediante el uso de la plataforma FTU2.

En conclusión, han sido comprobadas las herramientas que FTU2 puede ofrecer en el estudio del efecto de la radiación ionizante en dispositivos electrónicos y en el software utilizado, como en este caso, el protocolo SpaceWire que actualmente es utilizado en misiones espaciales donde son amplios los requerimientos para la mitigación de los efectos de la radiación cósmica.

7 CONCLUSIONES Y TRABAJOS FUTUROS

En conclusión, este trabajo presenta el diseño de un enlace SpaceWire para su posterior implementación en la plataforma FT-UNSHADES2 para simular los efectos de un entorno ionizante sobre dicho diseño. A partir de los resultados entregados por la plataforma se realizó una clasificación de los errores producidos en el enlace.

Se comenzó describiendo el entorno de radiación que puede encontrarse en el espacio, las partículas que lo componen y de donde provienen, con el objetivo de que se tenga conocimiento sobre un fenómeno invisible para el ojo humano pero que puede afectar en gran medida a las misiones espaciales y en menor medida a la tierra y sus habitantes. Se procedio con la descripción de los efectos de dicha radiación sobre los dispositivos electrónicos utilizados en el espacio, efectos temporales o permanentes, que pueden llevar a la perdida de un pixel en una imagen o, en un caso extremo y muy raro, a una catástrofe. Se continuo con la explicación de métodos para la mejora de la mitigación de los efectos producidos por la radiación ionizante, métodos necesarios debido a que la mitigación de estos efectos no solo puede depender de la mejora de los materiales utilizados para la electrónica. Para finalizar el capitulo, se procedió a describir un método y la herramienta CREME96 para predecir los efectos de la radiación ionizante en determinadas condiciones.

En el siguiente capitulo se explicó el funcionamiento del protocolo SpaceWire, el cual es un estándar de comunicación en dispositivos electrónicos utilizados a bordo en misiones espaciales debido a su gran velocidad de transmisión, fácil implementación y buena detección de errores. Se continúo dando ejemplos de misiones espaciales en las que SpaceWire es utilizado o esta planeado utilizarse. Concluyendo el capitulo se describieron otros tipos de comunicación utilizados en el espacio, como SoCWire que implementa ciertas mejoras en SpaceWire para la conformación de una arquitectura Network-on-Chip.

Se detalló el funcionamiento de la plataforma FT-UNSHADES2, sus modos de trabajo y sus herramientas para crear un entorno de radiación ionizante que permite analizar como responden los diseños creados por los usuarios ante dicha radiación para poder implementar mejoras, si son necesarias. Una herramienta fundamental para este trabajo.

Se continuó con el procedimiento seguido para el diseño en VHDL del enlace SpaceWire, los bloques que lo conforman y su conexión. Se procedió a explicar el funcionamiento y propósito de dichos bloques, los datos elegidos para transmitir en el enlace y la preparación previa a la implementación a FTU2.

En el ultimo capítulo se presenta la verificación del correcto funcionamiento del enlace SpaceWire en el modo debug de FTU2, para posteriormente realizar la campaña de inyecciones. Los resultados de la campaña de inyecciones de fallos devueltos por la plataforma fueron tratados mediante un script en Matlab para crear una clasificación de errores producidos en el enlace.

Se analizaron dos etapas, una etapa de inizialicacion del protocolo y la etapa de transmisión de datos. En el primer caso, un 82,8% de las inyecciones de fallos no influyeron en el funcionamiento del protocolo, mientras que, en la segunda etapa, un 77.35% de las inyecciones de fallos no llevaron a un error en el protocolo. No se ha realizado una predicción de los SEUs producidos para apoyar en la decisión de si estos resultados son buenos o no, pero sabiendo que en la FPGA Virtex-5QV [24], la cual esta cualificada para operar en el espacio, se producen $3.8e^{-10}$ errores/bit/día por lo que, aún utilizando todos los bits disponibles de la tarjeta, la cantidad de SEUs que se producirán será baja. Además, a partir de los resultados, menos del 50% de los SEU que se produzcan conllevarán a un error en el enlace, por lo que puede considerarse un buen resultado. Además, la mayor cantidad de los errores producidos son detectados por lo el enlace se puede reiniciar y

volver a funcionar normalmente.

Se ha comprobado que las herramientas aportadas por la plataforma FTU2 son de gran utilidad para el estudio del efecto de entornos de radiación ionizante sobre dispositivos electrónicos.

La necesidad de respuestas y conocimiento no acabará nunca, por ello las misiones espaciales tendrán objetivos cada vez más ambiciosos y, en consecuencia, la necesidad de nuevas tecnologías que introduzcan mejoras en el procesamiento de datos, aumento de la capacidad de almacenamiento, será algo necesario. Las nuevas tecnologías pueden traer consigo muchas ventajas, pero pueden tener desventajas que las más antiguas no tenían y por lo tanto nuevas investigaciones tendrán que ser realizadas para superar dichas desventajas. Nuevos protocolos de comunicación pueden ser desarrollados, los cuales sean más eficientes y más resistentes a la radiación ionizante.

REFERENCIAS

[1] "Radiation Hazards to Crews of Interplanetary Missions: Biological Issues and Reseatch Strategies", National Research Council Staff, 1997

[2] J. Barth, C. Dyer, and E. Stassinopoulos, "Space, atmospheric, and terrestrial radiation environments", Nuclear Science, IEEE Transactions on, vol. 50, no. 3, pp. 466-482, 2003.

[3] https://www.nasa.gov/sites/default/files/styles/226xvariable_height/public/599326main1_ssn_yearly-226.jpg?itok=2AlIcqmc

[4] https://www.nasa.gov/sites/default/files/styles/full_width/public/thumbnails/image/van_allen_probes_discov_new_rad_belt_cal.jpg?itok=UzTaHfhJ

[5] J.W. Howard, Jr, D.M. Hardage, "Spacecraft Environments Interactions: Space Radiation and Its Effects on Electronic Systems", MSFC, Alabama 35812, July 1999

[6] Michael Gordon, Ken Rodbell, "Single-Event Upsets and Microelectronics", IBM TJ Watson Research Center, 2015

[7] Dan M. Fleetwood, Ronald Donald, "Radiation Effects and Soft Errors in Integrated Circuits and Electronic Devices", 2004

[8] [En línea] https://radhome.gsfc.nasa.gov/radhome/papers/seespec.htm

[9] [En línea] https://radhome.gsfc.nasa.gov/radhome/papers/seeca6.htm

[10] Aaron Gerald Stoddard, "Configuration Scrubbing Architectures for HighReliability FPGA Systems", Brigham Young University, December 2015

[11] J. D. Engel, K. S. Morgan, M. J. Wirthlin, and P. S. Graham, "Predicting on-orbit static single event upset rates in xilinx virtex FPGAs", Los Alamos National Laboratory, Tech. Rep., 2006.

[12] [En línea] https://creme.isde.vanderbilt.edu/

[13] [En línea] http://spacewire.esa.int/content/Home/HomeIntro.php

[14] "SpaceWire – Links, nodes, routers and networks (ECSS-E-50-12C)", ESA – ESTEC, Tech. Rep., 2003

[15] [En línea] https://www.esa.int/Our_Activities/Space_Science/ExoMars/ExoMars_2020_rover

[16] [En línea] http://www.esa.int/ESA_Multimedia/Images/2017/03/ExoMars_rover2

[17] [En línea] https://www.esa.int/Our_Activities/Space_Science/BepiColombo_overview2

[18] [En línea] https://directory.eoportal.org/web/eoportal/satellite-missions/content/-/article/bepicolombo

[19] https://www.esa.int/var/esa/storage/images/esa_multimedia/images/2017/04/bepicolombo_in_cruise_configuration/16896735-2-eng-GB/BepiColombo_in_cruise_configuration_node_full_image_2.png

[20] Robert Gallager, "Low-Density Parity-Check Codes", 1963.

[21] B. Osterloh, H. Michalik, B. Fiethe, “Socwire: a SpaceWire Inspired Fault Tolerant Network-on-Chip for Reconfigurable System-on-Chip Designs in Space Applications”, International SpaceWire Conference, 2008.

[22] "Proximity-1 Space Link Protocol—Rationale, Architecture, and Scenarios", CCSDS, 2013.

[23] [En línea] http://ftu.us.es/

[24] Xilinx, “Radiation-Hardened, Space-Grade Virtex-5QV Family Data Sheet: Overview DS192 (v1.6)”, January 11, 2018